RL-TR-96-52
Final Technical Report
May 1996

# NCTI SIMULATION AND MODELING II

Decision-Science Applications, Inc.

Daniel Croghan

*APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.*



**Rome Laboratory**
**Air Force Materiel Command**
**Rome, New York**

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be releasable to the general public, including foreign nations.

RL-TR- 96-52 has been reviewed and is approved for publication.

APPROVED:

ALEX F. SISTI
Project Engineer

FOR THE COMMANDER:

JOSEPH CAMERA
Technical Director
Intelligence & Reconnaissance Directorate

# REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188



| 1. AGENCY USE ONLY (Leave Blank) | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | May 1996 | Final Jul 93 - Sep 95 |

**4. TITLE AND SUBTITLE**

NCTI SIMULATION AND MODELING II

**5. FUNDING NUMBERS**

C - F30602-93-C-0066
PE - 62702F
PR - 4594
TA - 93
WU - 1N

**6. AUTHOR(S)**

Daniel Croghan

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Decision-Science Applications, Inc.
1110 N. Glebe Road, Suite 400
Arlington VA 22201

**8. PERFORMING ORGANIZATION REPORT NUMBER**

N/A

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

Rome Laboratory/IRAE
32 Hangar Rd
Rome NY 13441-4114

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

RL-TR-96-52

**11. SUPPLEMENTARY NOTES**

Rome Laboratory Project Engineer: Alex F. Sisti/IRAE/(315) 330-4518

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**

Approved for public release; distribution unlimited.

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** (Maximum 200 words)

The overall goal of the Air Force's Non-Cooperative Target Identification (NCTI) program is to assess the feasibility and benefits of improving airborne long range target identification. Better target identification will contribute to improved weapons management, allow employment of beyond-visual-range (BVR) weapons, improve identification of friend or foe (IFF) or neutral, reduce fratricide, and improve situation awareness and air battle management.

This program enhanced the existing RL/IRA TAC BRAWLER-based NCTI modeling environment through further developing an existing "software Backplane" prototype, and continued the acquisition and integration of existing stand alone models into that environment. The enhanced capability allows for more accurate and realistic simulation studies to be undertaken for the purpose of assessing the feasibility and benefits of improving our aircraft's long range target identification capabilities.

**14. SUBJECT TERMS**

NCTI, TAC BRAWLER, "Software Backplane", Modeling and simulation

**15. NUMBER OF PAGES**

28

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | UL |

Table of Contents

# List of Figures

## 1.0 INTRODUCTION

This document reports on the performance of Decision Science Applications, Inc. (DSA) and Litton AMECOM under Department of Defense contract F30602-93-C-0066 to Rome Laboratory IRAE, titled NCTI Simulation Mod II.

The overall goal of the Air Force's Non-Cooperative Target Identification (NCTI) program is to assess the feasibility and benefits of improving airborne long range target identification. Better target identification will contribute to improved weapons management, allow employment of beyond-visual-range (BVR) weapons, improve identification of friend or foe (IFF) or neutral, reduce fratricide, and improve situation awareness and air battle management.

This program enhanced the existing RL/IRA TAC BRAWLER-based NCTI modeling environment through further developing an existing "software Backplane" prototype, and continued the acquisition and integration of existing standalone models into that environment. The enhanced capability allows for more accurate and realistic simulation studies to be undertaken for the purpose of assessing the feasibility and benefits of improving our aircraft's long range target identification capabilities.

There is also an on-going cooperative ESC/Rome Laboratory initiative to model the effects of NCTI equipment and techniques. Simulation activities at ESC/XR's Modeling, Analysis and Simulation Center (MASC) involve NCTI solutions at the theater level, while Rome Laboratory (RL) models proposed advances to existing equipment and techniques. Ultimately, RL develops and configures models of those advance capabilities for transition to ESC/XR. This effort is one of the stepping stones toward that transition.

The project included a survey phase to identify appropriate models, a design phase to define interface specifications for the integration of those models, and a development phase in which the augmented simulation environment is implemented.

## 2.0 NCTI SURVEY

The NCTI model survey identified, described, and evaluated stand-alone engineering models for integration into RL's TAC BRAWLER-based NCTI modeling environment. The results of the survey are available in DSA Report 84-1473. Of interest were models which contribute to the realistic and accurate simulation and evaluation of NCTI equipment and techniques. The survey began with an initial identification of potential model sources. Likely sources included Government and non-profit research laboratories, sensor contractors, and modeling and/or study houses.

This initial list was constructed from contacts known to RL from previous and current efforts and from contacts known to the DSA/Litton team from the team's broad base of corporate and individual work experiences. A limited, ad hoc document search provided additional contacts. In particular, other survey reports, such as the FOURMOSST report, expanded the initial list as well as identifying and describing particular models. Recent

symposia proceedings were also helpful. In addition, use was made of the Defense Technical Information Center (DTIC), and of other survey reports done for the U. S. Government in recent years. The latency date for survey data was established at 1985. A DTIC keyword search was also made to uncover any relevant reports in the DTIC database.

Sources were targeted according to their likely relevant sensor technology areas, i.e., radar, bistatic radar, radar signal modulation, high-resolution radar, ESM, acoustics, IR, EO, or data links. For example, in looking for advanced radar models, sources likely to be involved in developing advanced radars were contacted. Some attempts were made to address total corporate capability as well but discussions were often highly focused on specific contribution areas.

The survey was done in two portions. The initial portions focused on identifying sources of sensor, environmental and fusion models that could possibly be of utility in addressing the problem of long-range target identification. Additionally, respondents were asked to estimate the utility of their software to the NCTI problem and the ease of integrating the software into a distributed simulation environment. Respondents were also asked if there were any other suppliers for models with NCTI potential that they were aware of.

For the second, in-depth portion of the survey, the most promising models were selected according to the following criteria:

- Applicability to the NCTI problem
- Software language
- Ownership of the software
- Level of aggregation
- Model users at RL or the MASC
- Contribution to an NCTI-capable demonstration architecture.

The last criterion represents the DSA Team's belief that the prototype Backplane system model library should be initially populated with a set of models that can demonstrate some NCTI capability or technology. This will enable the development of the Backplane software and engagement model to be tied directly to a specific NCTI technology. The prototype system will then have a real, if somewhat limited, utility that can be demonstrated by RL to potential customers.

While the survey's scope and depth necessarily were limited by available funding and time, it considered a broad array of model sources. Scope may have suffered in that the survey was not systematically organized an because typically only one or two distinct groups in each organization were reached. Depth varied with the relevance of the potential modeling contribution. Additionally, most initial survey discussions took place in an

unclassified mode. Subjects requiring classified discussions may lack detail unless subsequent classified meetings or secure telephone calls were arranged.

## 3.0 BACKPLANE SOFTWARE DEVELOPMENT

The NCTI Backplane was developed as a proof-of-concept system under the NCTI Simulation Mod I project. The Backplane architecture draws from a computer hardware paradigm. The Backplane operates like a system bus, using software buffer circuits to define the bus architecture and system timing. This supports a loose confederation of models, supplying a generalized integration environment. The Backplane does not impose constraints restricting an analysis system built of independent models. The user is free to choose the coordinate system, timing interval, and the type definitions of the players in the confederation.

### 3.1 BACKPLANE FUNDAMENTALS

Figure 1 below illustrates the general concept of the NCTI Backplane system. The Backplane maintains a common representation for all connected models of the simulation ground truth, the Ground Truth Database (GTDB). Ground truth entities consist of platforms (aircraft or ground-based) and emitters. Emitters need not be associated with a platform in the ground truth. The Backplane also maintains a representation of the perceived situation for data fusion models, called the Perceived Truth Database (PTDB).

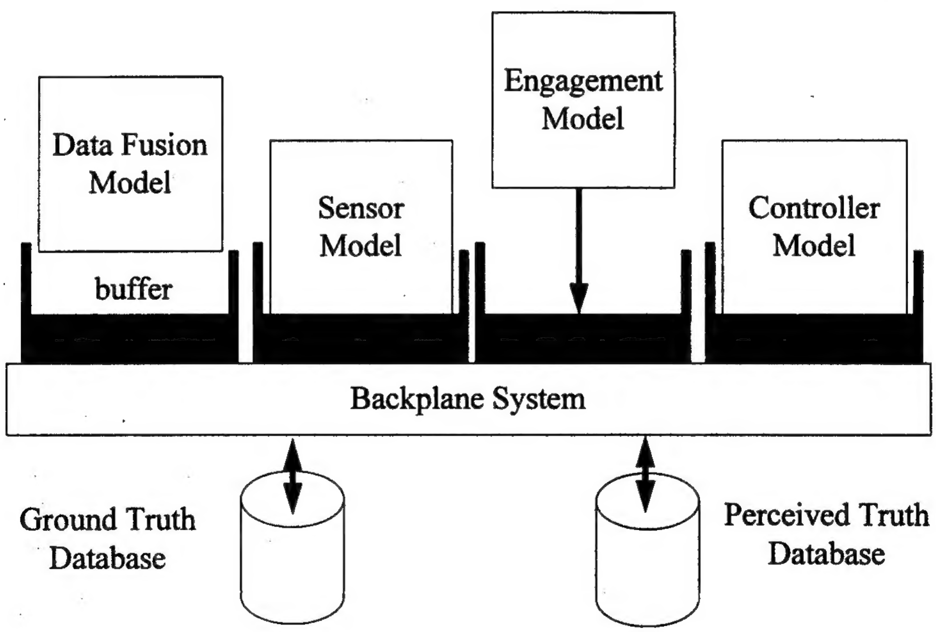


**Figure 1 NCTI Backplane Concept**

Any NCTI Backplane system consists of one or more engagement models, sensor models, data fusion models, and controller models. The engagement models are responsible for populating the GTDB with platforms and emitters, accepting Command and Control information from controller models, and resolving air combat engagements in the simulation. The engagement model provides a realistic environment and effectiveness Measures of Performance to analyze NCTI sensor models.

The sensor models in the Backplane can range from engineering level simulations to highly aggregated models of NCTI capable sensors. The models scan the GTDB and output detections to the data fusion model(s). Records in the GTDB provide enough basic information to drive aggregated models. More detailed models would be responsible for "expanding" the content of the GTDB record for internal use. These expansions could be based on a stochastic "mini-model," or could use the status information in the record to built up a more detailed record.

Data fusion models in the backplane take as input detections from sensor models, and interact with the PTDB to create a perception of the ground truth. The PTDB can contain false or merged tracks as well as perfectly correlated tracks. Representation of the errors in a track must be maintained internally in the fusion model. In this sense, the PTDB is a subset of the perception maintained by the data fusion system.

The controller models provide the command and control function to the Backplane that an Airborne Warning And Control System (AWACS) aircraft would in a real engagement. The controller model scans the PTDB, and based on the information present there, assigns fighters in the engagement model to fly vectors. The controller model also provides ID information on unknown/hostile aircraft to the fighters. This is the path through which the NCTI sensor capabilities effect the air combat engagement.

## 3.2 SOFTWARE IMPROVEMENTS

DSA made several considerable improvements in the proof-of-concept Backplane software developed under NCTI Simulation Mod I. First, the Backplane's C source code was cleaned up and streamlined. The source was converted to ANSI C standards, and prototypes were created for all Backplane functions. The directory structure used to maintain and run the backplane was improved and streamlined. Buffer circuit object code was moved out of individual directories and placed into archive files. Unused or commented out code was removed.

The Backplane had relied on INET (disk file) sockets for communications. This restricted the system to operate only on file-shared networks. DSA converted the INET socket interface to a TCP/IP socket interface, allowing the system to operate on a heterogeneous network which could be local, the Defense Simulation Internet, or even the Internet (for unclassified operations). The UNIX standard XDR (external data representation) software package was used to pack C structures into bit streams. This required a complete overhaul of the underlying message structure that the Backplane uses to pass messages from the buffer circuits to the main Backplane process. The XDR re-

implementation aids in maintenance, and later porting of the Backplane to UNIX systems other than Silicon Graphics.

Initially, the Backplane required only the engagement model to send a stop tick. DSA modified the Backplane to require stop ticks from all models specified in the input data file **config.dat**, allowing the Backplane to proceeded no faster than the slowest specified model. Other models may join (or leave) the simulation at any time. Several logic errors in the main Backplane function were discovered and fixed. These had mainly to do with operation of the controller-fighter messages.

Modifications to the Backplane buffer circuits include creating function prototypes for use by model integrators. Several improvements were also made to individual buffer circuit functions. These improvements supported integration of sensor models that do not have an internal platform model.

DSA was unable to get the Backplane viewer *colordraw* to work on the SGI implementation of X-windows using a Motif window manager since many SUN specific features had been utilized in developing the system. Because this system was for replays only, and the INET socket created by the Backplane was left intact, DSA assumes that *colordraw* still works on SUN systems.

## 4.0 ESM MODEL DEVELOPMENT

Access to the Backplane is provided by a well-defined family of software routines, called buffer circuits. These circuits provide an application-level interface into the Backplane environment, hiding implementation details and easing the task of integrating models. This functionality can be exploited to lower development time for new models and to speed modification of existing models. For users requiring additional information on the NCTI Backplane, please refer to the NCTI Backplane Software Design Document (SDD).

The model can be thought of as a single-threaded, non-adaptive ESM system, as shown in Figure 2. The electronic environment is provided by the Backplane buffer circuits. The receiver section is modeled as having certain sensitivity and observability characteristics. The sensitivity in this model is treated deterministically

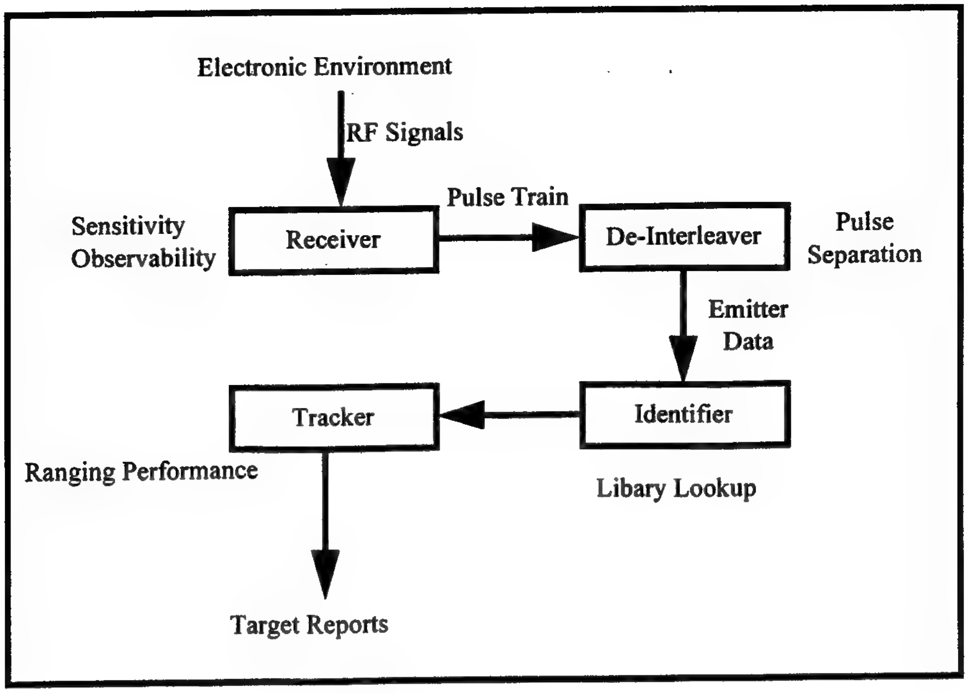


**Figure 2 ESM System Model**

as a function of the emitter characteristics: at a particular slant range from the ESM system, an emission can be detected unconditionally. Observability is represented in two domains: spatial and frequency. The spatial domain is determined by the relative geometry between the host platform and emitter to determine if an ESM system antenna can "see" the emitter. The pointing of the emitter antenna is used to determine if the intercept will be a mainlobe or sidelobe observation. Whether the receiver happened to be observing the frequency band of the emitter during the illumination time is a matter of the internal state of the ESM system which is modeled stochastically as a function of a bandwidth parameter known as the revisit time.

In physical terms, one can next think of a "pulse train," represented by a number of emitters passing through the receiver, being sent to a de-interleaving processor. This processor attempts to sort out the individual pulses belonging to a particular emitter from all of the other emissions in the pulse train. This system is modeled stochastically with the probability of de-interleaving being an function of the number of other emitters within the pulse train.

If the emitter is successfully de-interleaved, emitter data is then sent to an identifier, which looks up the emission characteristics and declares an identification of the emitter. In this model, we assume that the identifier works perfectly.

The final step in the ESM system process is the tracker. This system is modeled as two single-dimension (azimuth and range) weighted average tracks. Ranging accuracy is modeled as a function of the number of time steps the emitter has been observed. The azimuth measurement errors are assumed to be Gaussian in distribution limited to within

3 standard deviations of the mean, and are characteristic of several factors. These are a function of the observing antenna, and are characteristic of the emitter bearing and center frequency.

The input data file for the ESM model is a text-based description of the capabilities of a particular ESM system that will be represented. The input file is divided into sections, each corresponding to an object in the ESM system. The beginning of a section is denoted by the name of the object, then the elements describing that object are listed. The section is closed with an "END" labeled line. The sections can appear in any order.

## 5.0 BRAWLER DEVELOPMENT

DSA designed and implemented an interface between Brawler and the NCTI Backplane system. The interface uses both ANSI C and ANSI FORTRAN 77 standard source code. Generally, the FORTRAN source code interfaces with Brawler internals (subroutines and common blocks) and the C source manipulates the NCTI buffer circuit functions. The interface developed for Brawler consists of two parts. The first part implements a long-term maintainable interface between Brawler and the NCTI backplane. This part is concerned with representing internal Brawler ground truth and a Command and Control interface between fighters and an external process. The second part uses the Brawler controller model to represent a controller within the Backplane. This is an expedient interface and enables a realistic controller model to be imbedded in the Backplane at low cost. The second interface is described in section 5.3 below.

**SOURCE CODE TRANSPORTABILITY**. The C and FORTRAN source integration relies on UNIX convention regarding the naming of relocatable objects and FORTRAN common blocks. The argument list integration relies on the ANSI standards for C and FORTRAN. DSA has found that the UNIX convention is supported by some non-UNIX operating systems such as VAX/VMS. The DEC/ULTRIX operating system uses a slightly different convention, however compiler switches can be used to support the VMS (and thus UNIX) mixed language protocols. The Brawler-Backplane interface should therefore be transportable to most UNIX, VMS and ULTRIX operating systems with a minimum of re-coding.

**BASIC INTERFACE.** The Brawler interface utilizes the buffer circuit library to export Brawler's internal representation of ground truth to the Backplane Ground Truth Database (GTDB). The elements of Brawler's ground truth exported are:

- *Aircraft and Missile states*: Position, velocity and status (alive or dead).
- *Emissions*: Emitter type, frequency and duration. Emitter types exported are aircraft and missile radars, communications radios, IFF, and jammers.

Command and Control can be provided to Brawler through several Backplane buffer circuits. The implemented buffer circuits include vectoring commands and hostile/friendly ID information. The interface controls the advancement of Brawler's internal simulation clock, synchronizing it with the Backplane (and thus other confederated models), through a rendezvous event. The interval between synchronization events is user-controllable through a data file.

## 5.1 CONNECTIONS INTERFACE

Prior to this project the Brawler confederation interfaces were purpose-built. There was a EADSIM interface, a prototype DIS interface, and a separate ALSP interface. The prospect of adding yet another interface type for the Backplane raised serious concerns about the maintainability of four separate interfaces. A connection-based interface design, using a data-neutral format and pointers to write and read functions promised to rationalize the interface code and greatly improve maintainability. NCTI project and Air Force Studies and Analyses Agency funds (for an improved EADSIM interface) were used to implement the connections interface.

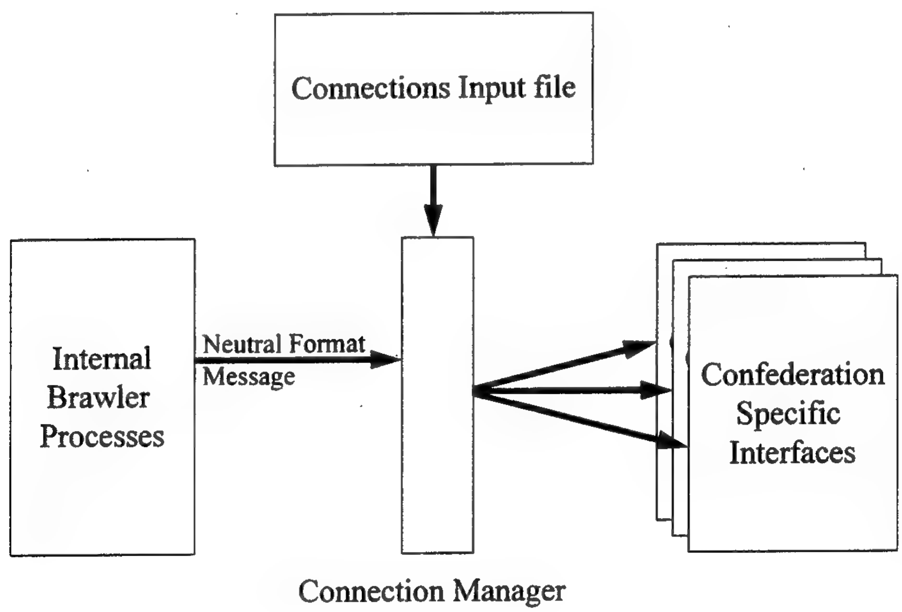


**Figure 3 Connections Interface**

Figure 3 illustrates the connections design. Various parts of Brawler source that support confederated operation produce a neutral-format message. DIS 2.03 PDU's form the basis for the neutral-format message, which is implemented as an ANSI C data structure. Operation of the connection is then straightforward. For example, the subroutine that integrates equations of motion for Brawler aircraft calls a confederation routine. This rou-

tine consults a dead-reckoned external representation of the aircraft's state. Dead-reckoned states are kept using a constant acceleration model of the aircraft state. If any of several parameters exceed the dead-reckoned representation, a subsidiary routine is called. This function creates a neutral format message that is passed to the connection interface.

The connection interface checks a C structure to see which confederations have been specified, and whether that confederation accepts messages of the input type. If both of checks pass, the interface passes control to an export function. The function is accessed through a function pointer that was assigned to the connections structure during initialization. The function takes as an argument a pointer to the neutral-format message, and is responsible for understanding how to translate this into a confederation-specific message. For a DIS interface, the structure must be converted to a specific PDU format, then packed into a bit stream. A socket interface then accepts the bit stream and sends it out to the network. In the case of the NCTI backplane, the structure maps onto the argument list of a specific buffer circuit function. The buffer circuit function then transmits the message to the Backplane.

For input into Brawler the process starts with confederation-specific input functions. These functions are called by the connection manager at defined intervals. For the NCTI backplane, the rendezvous event calls the read functions. The read function converts the input data into a data-neutral C structure. The function then calls a generic interface within Brawler that knows how to update Brawler's internal state to reflect the information contained in the message from an external process. The specific Backplane functions are listed in Table 1:

**Table 1 Buffer Circuit/Message Cross Reference**

| Buffer Circuit Call | DIS PDU Analog | Special Actions |
|---|---|---|
| declare_combatant_emitter | ElectromagneticPDU | Emitter has no Backplane ID yet |
| update_combatant_emitter | ElectromagneticPDU | Emitter has Backplane ID in cross-reference |
| declare_combatant | EntityStatePDU | Entity has no Backplane ID yet |
| update_combatant | EntityStatePDU | Entity has Backplane ID in cross-reference |
| get_vector | SignalPDU | Vector is expanded to meet Brawler format |
| get_report_airframe | SignalPDU | Airframe in report must exist in Brawler |
| synchronize_combatant | EndOfDataPDU | Experimental |

The messages implement an interface between fighter aircraft modeled in Brawler and an external Command and Control process that vectors the fighters and identifies threat aircraft at long range.

## 5.2 FIGHTER-BACKPLANE INTERFACE

The Brawler fighter-NCTI controller interface supports vectoring information and ID of threat aircraft at long ranges. The interface uses two message types: a vectoring message and an aircraft ID message. Receipt of a vectoring message by the flight leader will cause the Brawler flight to turn toward the desired heading. Brawler aircraft will continue to fly the desired heading until the message times out. The default time out is 1000 seconds. A slight disconnect exists between the Backplane representation of Command and Control and Brawler's. Brawler expects to receive a commanded target from a controller when flying a vector, however the Backplane does not support this message type. If the Brawler pilots do not receive a commanded target, they will continue to fly the vector, and not engage nearby hostiles. While this can initially prove frustrating to the analyst, using the production rule facility of Brawler provides an easy remedy. The analyst simply specifies what conditions (such as range to unknown/hostile aircraft) will cause the pilot to take control of the intercept. In tactical terms, this is the equivalent of a "Judy" call from the pilot to the controller.

Receipt of an aircraft ID message will cause the Brawler pilot to update his perception of the target, including ID if it is included in the message. If the pilot was not aware of the target, the Backplane message causes him to add the target to his mental model. An analyst can then study the effect of NCTI technologies by assigning pilots headings to fly, and informing the fighters of targets, and their identity.

To further aid the analyst in quantifying the effect of long-range NCTI, a new type of Rule of Engagement (ROE) parameter was developed for the NCTI Backplane. This ROE is called C2_ID and specified in the flight description section of the Brawler **SCNRIO** file. The ROE constrains the Brawler pilots not to shoot threat aircraft unless a Command and Control message specifying a hostile ID is received. A visual ID of the hostile aircraft overrides this constraint.

## 5.3 BRAWLER CONTROLLER-BACKPLANE INTERFACE

Illustrating the full potential of the NCTI Backplane system requires a Command and Control model, directing the fighters and providing target ID information. Rather than develop an entirely new controller model, DSA re-used the existing Brawler controller. This re-use also illustrates the flexibility of the Backplane architecture. Brawler participates in the demonstration Backplane system as the engagement model and simultaneously as the controller model. Figure 4 shows how the controller interaction with the Backplane is isolated from the integration as an engagement model.

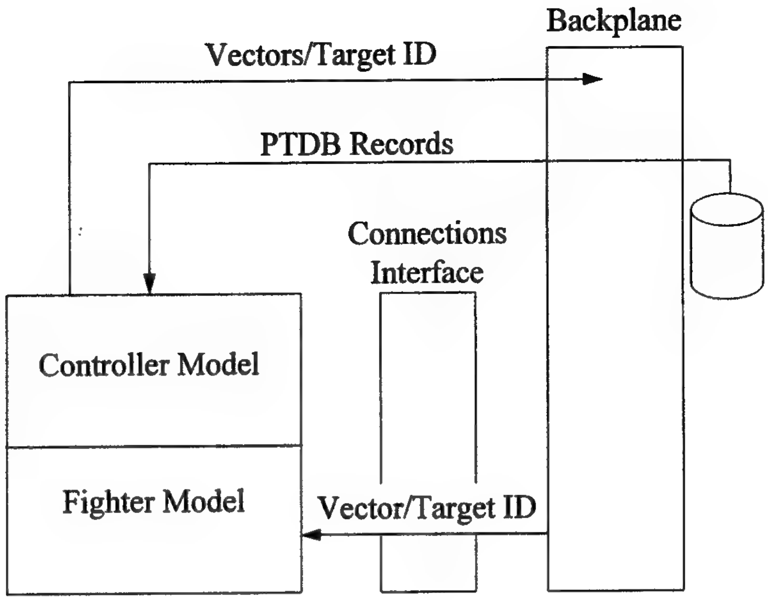


**Figure 4 Controller-Fighter Integrations**

The Brawler controller uses a radar internally modeled in Brawler, and a Backplane NCTI ESM sensor. By correlating ID information from the Backplane with Brawler radar tracks, the controller creates a fused picture of the air battle. Vectoring and target information channels through an NCTI communications interface to the Brawler fighters. By not declaring the controller model, the Backplane does not require the controller to synchronize, enabling the engagement portion of Brawler to provide synchronization for both participants.

Since the controller model was integrated to demonstrate the potential of the Backplane system, DSA integrated the Brawler controller directly through the NCTI buffer circuits. Bypassing the connections interface protocol simplified the controller integration, and enabled Brawler to synchronize just once, as an engagement model.

Using the Brawler controller also solved the problem of modeling the command and control relationship between Brawler fighters and an external controller. The relationship already exists in Brawler, but at this time cannot really be modeled with the Backplane system.

## 6.0 DEMONSTRATION

DSA successfully demonstrated an NCTI Backplane system operating with:

- An engagement model (Brawler)

- A NCTI-capable long-range sensor (ESM model)
- A Data Fusion Model
- A Controller Model (Brawler as a controller)

This demonstration is available at RL/IR's ICARUS facility and on DSA's Arlington, VA unclassified network. Graphical replay of the engagement is provided by the Brawler X-windows post-processor *grmain*. A schematic of demonstration system is shown in Figure 5. Of the components in the figure, only the fusion model has not been described. Since a fusion system is required for proper operation of the NCTI Backplane one must be present. However, the fusion model can be simple or complex. In this case, the fusion model is a simple "bent pipe," that takes detections from the ESM model and routes them to the Perceived Truth Database. The Brawler controller model then fuses ID information from the Perceived Truth Database (PTDB).

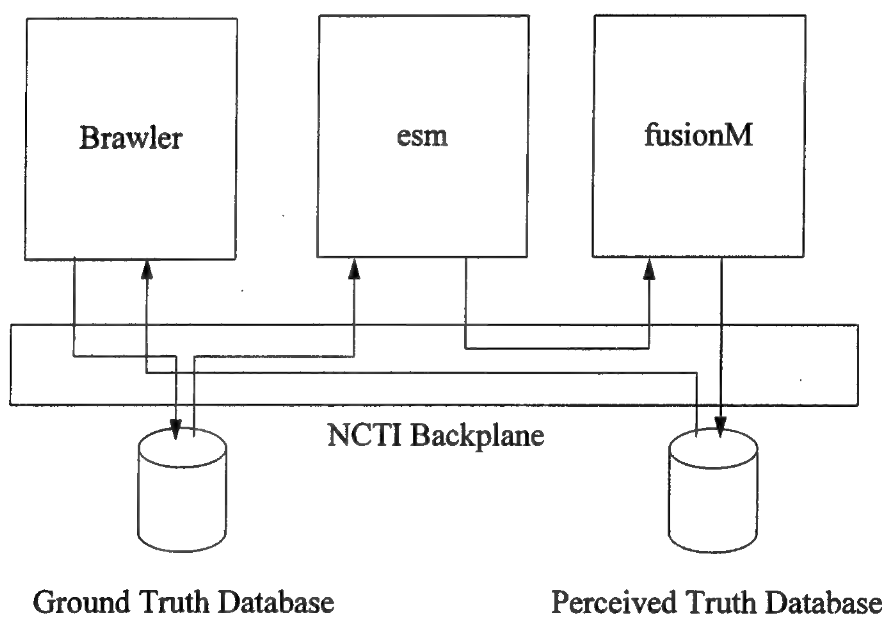


**Figure 5 Demonstration System**

### 6.1 SYSTEM REQUIREMENTS

This demonstration system was compiled using following resources:

- GNU 2.6.3 C++ compiler (for the ESM model)
- IRIX 5.3 C compiler (for the Backplane, buffer circuits and Brawler C code)

- IRIX 5.3 FORTRAN compiler (for Brawler source)

The demonstration system runs on a Silicon Graphics Indigo II workstation and is data driven in important network parameters such as IP address and port number. Porting the system to another UNIX platform (such as SUN/SOLARIS) would be greatly eased by using an ANSI C compiler for the C source code, as well as the appropriate GNU C++ compiler. The total disk space required for hosting the demonstration is about 80 megabytes for the NCTI system and 16 megabytes for Brawler.

## 6.2 NCTI SCENARIO

The scenario chosen for the demonstration pits two Blue fighters (under the control of an AWACS) against a Red strike force, consisting of bomber flight, preceded by escorting fighters. Figure 6 shows the opening set-up. The Blue fighters and AWACS begin the simulation orbiting cap stations. As the hostile strike force is detected, the AWACS vectors the Blue fighters against the unknown targets, or bogies. When the AWACS controller identifies the bogies as hostiles, the controller sends "bandit" calls to the Blue fighters. If the fighters gain radar contact with unknowns or hostiles, they will take over the intercept at 20 nautical miles distance. They must close to visual range to identify a bogie, or unknown. However, if they have received a bandit ID on the target, the fighters may employ their Beyond Visual Range (BVR) missiles.

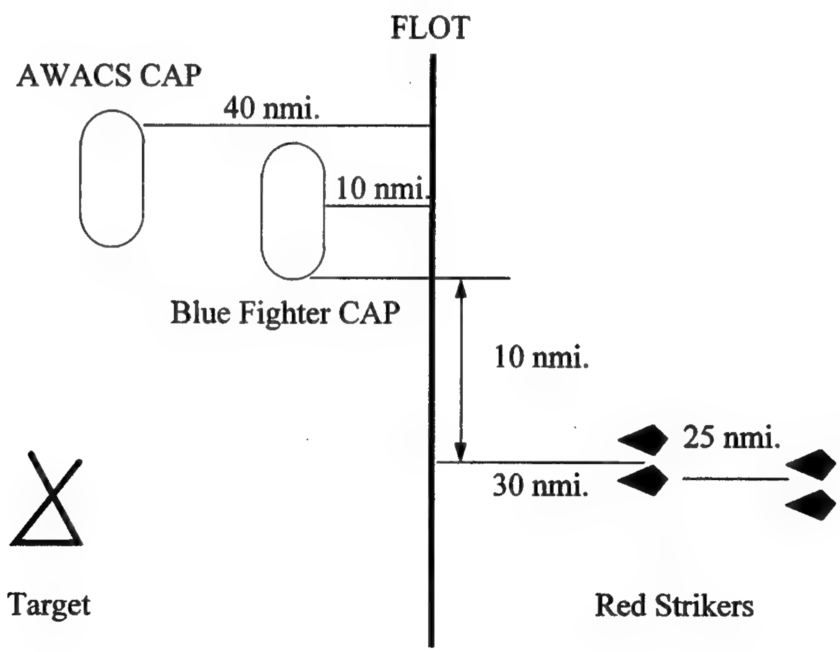


**Figure 6 Air Combat Scenario Set-up**

## 6.3 BACKPLANE CONFIGURATION

For this demonstration, the NCTI Backplane system was configured as shown in Figure 5. This configuration is listed in the file **config.dat** in the /app/ncti/run directory. Brawler serves as the engagement model, and provides the time interval definition for the system. This model must be run manually after the Backplane has been started with the script *run_backbone*. After Brawler is running, the backplane will automatically start the ESM sensor model *esm* and the data fusion "bent pipe" model, *fusionM*. Each of the models listed in the configuration file must rendezvous by sending stop ticks (by calling the appropriate synchronize buffer circuit). Once all the listed models have sent a stop tick, control is released back to the individual models by the buffer circuits.

Brawler will create another model on the system if a GCI/AWACS controller is specified in the **SCNRIO** file as being an "NCTI" controller. This connection to the backplane does not have to synchronize because it is not listed in the configuration file. Brawler as the engagement model provides synchronization.

Entity and Emitter type cross reference information must be manually controlled by the user. Brawler uses the "extra" portion of a DIS EntityKind structure as the single number used by the Backplane to specify the type of an entity. Emitter types are a mangled identifier consisting of 100 times the DIS system function index plus the DIS system name index. The ESM model input file has corresponding type definitions with those used by Brawler.

## 6.4 MODELING ASSUMPTIONS

Several important assumptions were made in developing the scenario, and in specifying the weapons systems and aircraft employed by both sides. First, the scenario must be unclassified, therefore all the weapons, radars and aircraft in the scenario consist of notional data sets with none of the Blue or Red systems representing any current real-life hardware. This is especially true of the missile and radar systems. All radars used in the scenario do not have a Track-While-Scan (TWS) capability and none of the missiles are of a command-guided (AMRAAM) type. Source code for both of these systems in Brawler is classified and therefore cannot be used. The is a limitation because unclassified missiles available in Brawler have launch ranges on the order of 10 nautical miles, at most, which barely represents a BVR capability. Additionally, each Blue fighter can target only one hostile at a time, because the "BVR" missiles require single-target-track (STT) illumination from the launching aircraft.

### 6.4.1 Backplane System

The backplane system is set up through the input file **config.dat** to require Brawler, the ESM model, and the fusion model to synchronize. The Backplane will wait until all three models have sent a stop tick before returning from the synchronize buffer circuit function. Additional models (such as the Brawler controller) may join the simulation without synchronizing. The Backplane has no inherent requirement for a synchronizing

interval, so the analyst is responsible for ensuring that all three models use the same three second interval. The Backplane also does not specify a coordinate system. In this case, the Brawler game-centered coordinate system is used, with the X axis pointing East, Y south and Z downward. All distances are measured in feet, velocity vectors are of unit length, and speeds are in feet/second.

### 6.4.2 ESM Model

The ESM model in the demonstration system has no defined location in the Backplane system. Instead, the model assumes that it is located on a host airframe in the GTDB. This airframe is identified by a unique type, specified in the input data set. Location and pointing information for the ESM system is derived from the airframe's GTDB record. The ESM model computes a main beam swept volume, assuming that the emitter pointing and scan rate data reported in the GTDB was current at the beginning of the time step. Range between the host airframe and the emitter are assumed to be static over the time step. This is reasonable for a long-range system.

The ESM model is data driven for its detection performance. For the demonstration scenario, a notional data set was drawn up using informed approximations as to the performance. Detection ranges for radar sidelobes were set to twice the nominal radar detection range, and for main-beam detections, four times the radar track range. The performance against real-world systems was taken from Jane's All The World's Weapons Systems, 1987 Edition. For the notional systems in Brawler, the performance was estimated using the radar data set.

### 6.4.3 Fusion Model

In the demonstration system, the fusion model exists merely to populate the PTDB with ESM detections. The actual ESM/radar fusion is done in the Brawler controller model. The fusion model assumes that detections are perfectly correlated. The previous detection of an emitter is overwritten by the current detection (if any), and there are no false targets in the PTDB.

### 6.4.4 Brawler Controller Model

The Brawler controller model makes several important assumptions about its existence in the NCTI Backplane system. First, the controller model will not detect targets using the Backplane PTDB. The PTDB detections provide ID information only. Second, the ID information is correlated to the controller radar, which is modeled in Brawler. If a PTDB target is within 1 nautical mile of a radar track, the ID information is assumed to belong to that radar track. Ties are resolved using the track closest to the ID observation. If a radar target is identified by the controller, a message is routed to each member of a flight under its control. The message is reported in the Brawler LOG file as a "clear to shoot" message. They are repeated at approximately 1 minute intervals, as Brawler pilots can lose confidence in a target's ID. Aircraft under control are specified within Brawler, using the **SCNRIO** input file. Third, messages from the fighters to the controller are routed

through Brawler's communications channels, not the Backplane's. This part of Brawler's command and control model does not effectively map onto the Backplane Buffer circuits.

### 6.4.5 Brawler Fighter Model

Controller to fighter communications in the NCTI backplane map onto the "vectoring" controller model in Brawler. Pilots accept vectors from the controller, fly the assigned heading and airspeeds, and use the ID messages to create and update tracks in their perception of the tactical situation. There are three important assumptions in the fighter integration. First, the originating aircraft or controller of a message must exist in Brawler's internal ground truth. This is because the message information is placed into Brawler's internal radio channel model. The originating platform can be ghosted, or owned by Brawler, but it must have been registered as being "alive," otherwise the message to the Brawler pilots will be deleted. The fighter pilots do not care from whom the message comes; it is enough that the message is on their internal "frequency" and they will respond appropriately.

Second, a part of the internal "vectoring" controller Command and Control structure does not exist. The "target location" message from the controller to the fighter is not available in the Backplane buffer circuits. This causes the Brawler pilots to ignore any unknowns or hostiles and continue to follow the assigned vector without engaging. This problem is easily handled through the production rule variable *ingngc4_flag*. When the pilot has radar contact on an unknown or hostile aircraft within 20 nautical miles, he ignores the vectoring information and takes over the intercept. The production rules provided in the demonstration system provide an example of how the analyst may tailor this decision as preferred.

Third, the Brawler pilot will infer the identity of an unknown if it is flying in close proximity to a known hostile, i.e. in "formation ." This behavior is not specific to the Backplane demonstration system and must be understood by an analyst using the system in another scenario.

## 7.0 SUGGESTIONS FOR FURTHER DEVELOPMENT

DSA believes that significant improvements are possible in the Backplane system. These result from our experience during the ESM model development and integration of Brawler into the Backplane.

### 7.1 BACKPLANE SOFTWARE

The first improvement suggested for the Backplane system is to increase the efficiency of the system. The Backplane is very slow, because the system uses less than 30 percent of the available CPU. DSA believes that this is because the message read utility in the main program loop polls the socket every time through the loop. Setting up an interrupt-driven read might greatly improve the CPU utilization rate.

Second, the Backplane system would benefit from a dynamic registration of synchronized models instead of the file-based system now used. This would allow models to request to be synchronized when they joined the Backplane simulation or request to be dropped from the system when ready to leave it. This dynamic registration would increase the flexibility of the system considerably.

A third suggestion is to modify the Backplane GTDB to contain model-assigned identifiers rather than having the Backplane assign identifiers. This would enable analysts to specify command and control hierarchies outside of individual engagement models. Specifying host platforms for sensor models would also be simplified.

### 7.2 ESM MODEL

The *em* model would be greatly improved by creating an input data set from an engineering level simulation. The current input data are derived from unclassified sources and at best, represent a notional ESM system.

The tracker function in the model is currently stubbed, and the model sends the raw, errored observations. A second improvement in the model would be to improve the tracker representation. The utility of this improvement depends on the type (simple or complex) of data fusion model that is hooked into the Backplane. A simple fusion model would benefit most from smoothed emitter tracks, whereas a complex model might require the raw observations.

### 7.3 FUSION MODEL

Data fusion is a complex and technically challenging field. Anything more than the simple model already extant in the Backplane would require a substantial development effort. The NCTI survey identified several programs involved in data fusion. In particular, the Mitre Fusion Evaluation Testbed program seems a likely candidate source for “real” fusion models for inclusion in the Backplane.

### 7.4 CONTROLLER MODEL

The controller model is adequate. Improvements could be made in breaking the controller model out of Brawler for operation as a separate process. Another possibility lies in using the controller model as a Human-In-The-Loop interface to the Backplane simulation. This interface could be used by experienced controllers to gain insight into the effectiveness of NCTI technologies.

### 7.5 BRAWLER

The Brawler interface is adequate for most uses of the NCTI Backplane system. Improvements to the interface should focus on a better representation of the controller-fighter interactions. Specifically, the message set and responses can be expanded to more closely resemble those found in real life. For example, Brawler fighter pilots can be

modified to *ask* for target information, if uncertainty exists in their perception of the situation.

Additional improvements can be made by modeling a tactical data link with more fidelity that currently exists in Brawler. This data link could be fed by a fusion model, or directly by the controller model in the Backplane.

☆U.S. GOVERNMENT PRINTING OFFICE: 1996-509-127-47055

## *MISSION*
## *OF*
## *ROME LABORATORY*

Mission. The mission of Rome Laboratory is to advance the science and technologies of command, control, communications and intelligence and to transition them into systems to meet customer needs. To achieve this, Rome Lab:

a. Conducts vigorous research, development and test programs in all applicable technologies;

b. Transitions technology to current and future systems to improve operational capability, readiness, and supportability;

c. Provides a full range of technical support to Air Force Materiel Command product centers and other Air Force organizations;

d. Promotes transfer of technology to the private sector;

e. Maintains leading edge technological expertise in the areas of surveillance, communications, command and control, intelligence, reliability science, electro-magnetic technology, photonics, signal processing, and computational science.

The thrust areas of technical competence include: Surveillance, Communications, Command and Control, Intelligence, Signal Processing, Computer Science and Technology, Electromagnetic Technology, Photonics and Reliability Sciences.